Operação		Instrução *assembly*	*Flags* afetadas				Descrição
Aritmética	Add	`ADD Rd, Rn, <operand2>`	N	Z	C	V	Rd := Rn + operand2
	with carry	`ADC Rd, Rn, Rm`	N	Z	C	V	Rd := Rn + Rm + Carry
	Subtract	`SUB Rd, Rn, <operand2>`	N	Z	C	V	Rd := Rn - operand2
	with carry	`SBC Rd, Rn, Rm`	N	Z	C	V	Rd := Rn - Rm - Carry
Lógica	AND	`AND Rd, Rn, Rm`	N	Z			Rd := Rn AND Rm
	OR	`ORR Rd, Rn, Rm`	N	Z			Rd := Rn OR Rm
	Exclusive-OR	`EOR Rd, Rn, Rm`	N	Z			Rd := Rn EOR Rm
Deslocamento	Logical shift left	`LSL Rd, Rn, #<immed_4>`	N	Z	C		Rd[15:immed_4] := Rn[15-immed_4:0]; Rd[immed_4-1:0] := 0
	Logical shift right	`LSR Rd, Rn, #<immed_4>`	N	Z	C		Rd[15-immed_4:0] := Rn[15:immed_4]; Rd[15:15-immed_4+1] := 0
	Arithmetic shift right	`ASR Rd, Rn, #<immed_4>`	N	Z	C		Rd[15-immed_4:0] := Rn[15:immed_4]; Rd[15:15-immed_4+1] := Rn[15]
	Rotate right	`ROR Rd, Rn, #<immed_4>`	N	Z	C		Rd[15-immed_4:0] := Rn[15:immed_4]; Rd[15:15-immed_4+1] := Rn[15:15-immed_4-1:0]
	Rotate right extended	`RRX Rd, Rn`	N	Z	C		Rd[14:0] := Rn[15:1]; Rd[15] := Carry; Carry := Rn[0]
Comparação	Compare	`CMP Rn, Rm`	N	Z	C	V	Realiza a operação Rn - Rm e atualiza as *flags* do CPSR
Transferência de dados entre registos	Move	`MOV Rd, <operand2L>`					Rd := operand2L
	NOT	`MVN Rd, Rm`	N	Z			Rd := 0xFFFF EOR Rm
	to higher byte	`MOVT Rd, #<immed_8>`					Rd[15:8] := immed_8
	and restore CPSR	`MOVS PC, LR`	N	Z	C	V	PC := LR; CPSR := SPSR
	register to PSR	`MSR PSR, Rm`					PSR := Rm (bits 0 a 5)
	PSR to register	`MRS Rd, PSR`					Rd := PSR
Transferência de dados com a memória	Load						
	Word	`LDR Rd, <a_mode1>`					Rd := [address]
	Byte	`LDRB Rd, <a_mode3>`					Rd[7:0] := [address]; Rd[15:8] := 0
	Store						
	Word	`STR Rd, <a_mode2>`					[address] := Rd
	Byte	`STRB Rd, <a_mode3>`					[address] := Rd[7:0]
Manipulação da Pilha	Push	`PUSH Rm`					SP := SP - 2; [SP] := Rm
	Pop	`POP Rd`					Rd := [SP]; SP := SP + 2
Controlo	Branch	`B{cond} label`					R15 := label
	with link	`BL label`					R14 := R15 + 2; R15 := label

Tabela 1 – Conjunto de instruções do P16.

Modo de Endereçamento 1 (<a_mode1>)			
Indireto		[Rn]	Equivalente a [Rn, #0]
Indexado	Com constante	[Rn, #<immed_4>]	address := Rn + immed_4[3:1]:0
	Por registo	[Rn, Rm]	address := Rn + Rm
Relativo		labelS	address := labelS

Tabela 2 – Modos de endereçamento possíveis para acesso à palavra (leitura).

Modo de Endereçamento 2 (<a_mode2>)			
Indireto		[Rn]	Equivalente a [Rn, #0]
Indexado	Com constante	[Rn, #<immed_4>]	address := Rn + immed_4[3:1]:0
	Por registo	[Rn, Rm]	address := Rn + Rm

Tabela 3 – Modos de endereçamento possíveis para acesso à palavra (escrita).

Modo de Endereçamento 3 (<a_mode3>)			
Indireto		[Rn]	Equivalente a [Rn, #0]
Indexado	Com constante	[Rn, #<immed_3>]	address := Rn + immed_3
	Por registo	[Rn, Rm]	address := Rn + Rm

Tabela 4 – Modos de endereçamento possíveis para acesso ao byte.

Operando	<operand2>	<operand2L>
Constante	#<immed_4>	#<immed_8>
Registo	Rm	Rm

Tabela 5 – Tipos possíveis para o segundo operando.

{cond}	Descrição
ZS / EQ	Zero Set / Equal
ZC / NE	Zero Clear / Not equal
CS / LO	Carry Set / Unsigned lower
CC / HS	Carry Clear / Unsigned higher or same
LT	Signed less than
GE	Signed greater than or equal

Tabela 6 – Condições de salto possíveis.

Bits dos registos PSR							
Reservado		M	I	N	V	C	Z
15	6	5	4	3	2	1	0

Tabela 7 – Campos dos registos PSR.

Legenda das tabelas:

Rd, Rm	Podem referenciar qualquer registo do banco de registos (R0 – R15).
Rn	Pode referenciar um dos registos da parte baixa do banco de registos (R0 – R7).
<operand2>	Ver a Tabela 5.
<operand2L>	Ver a Tabela 5.
<immed_n>	Uma constante codificada com n-bits na própria instrução usando o código binário natural.
<offset_n>	Uma constante codificada com n-bits na própria instrução usando o código binário dos complementos.
<PSR>	Pode referenciar o *Current Processor Status Register* (CPSR) ou o *Saved Processor Status Register* (SPSR).
<a_mode1>	Ver a Tabela 2.
<a_mode2>	Ver a Tabela 3.
<a_mode3>	Ver a Tabela 4.
{cond}	Ver a Tabela 6. Omitir no caso de saltos incondicionais.
label	Deve referenciar um endereço na vizinhança de ±1 KB da instrução em causa.
labelS	Deve referenciar um endereço na vizinhança de +128 B da instrução em causa.